COBB

INSIDE DOS

Tips & techniques for MS-DOS & PC-DOS

October 1992
Vol. 3 No. 10

Creating a simple menu system

One of the most common complaints about DOS is that it lacks a "user-friendly" interface. Granted, the command prompt can seem mysterious when you don't know the path and name of the program file you want to run. However, you don't need to settle for the uninformative C:\> prompt when you boot your system. Instead, you can tell DOS to display an informational menu. Then you—or another user—will have to enter just the command shown on the menu to run the corresponding program or set of commands.

In this article, we'll show you how to create a basic menu system using two batch files. The first batch file, EASYMENU.BAT, displays a menu listing your choices. The menu will tell you to type *go* followed by the number or name you've assigned to the menu selection to run the application. Typing *go* calls GO.BAT, the second component of the menu system. GO.BAT's labels and GOTO commands let you combine into one batch file the instructions for running all your menu choices, which saves space on your hard disk. Unlike some menu programs, you can create EASYMENU.BAT and GO.BAT with simple DOS commands. (Other menu systems require you to create a DEBUG script to capture the user's input.)

Creating EASYMENU.BAT

We suggest you begin by creating EASYMENU.BAT. Creating the menu first will help you outline the structure of GO.BAT. Figure A shows the version of EASYMENU.BAT we'll use in our example. As you can see, this batch file clears the screen, then echoes the lines to the screen to create the menu. Finally, EASYMENU.BAT sets your system prompt to the phrase *Enter your choice*, followed by a colon.

Of course, you can substitute the tasks you most commonly perform for those in our sample EASYMENU.BAT. As you can see, you needn't limit yourself to program names for the descriptions on the menu. For example, *Create/edit a document* runs Microsoft Word. We've also included two commands for formatting 5.25" diskettes, as well as an option to quit the menu. We encourage you to use the menu to run all your routine tasks.

Figure A

```
@echo off
cls
echo.
echo.
echo.
echo.
echo.
echo.
echo.
echo          To do this:                  Enter:
echo.
echo          Create/edit a document       go 1
echo          Format a 1.2 Mb diskette     go 2
echo          Format a 360 Kb diskette     go 3
echo          Quit the menu                go 4
echo.
echo.
prompt [Ctrl]P[Alt][255]  Enter your choice:
```

You can adapt EASYMENU.BAT to display your favorite applications and the codes to run them.

As always, you'll need to create the batch file using the DOS 5 Editor or a word processor that can save files in an ASCII, or text-only, format. In our example, we'll use the DOS 5 Editor, since the MS-DOS SETUP program automatically installs EDIT.EXE in your DOS directory. (Remember, if you use a word processor file format that

IN THIS ISSUE

allows you to insert formatting codes, the codes will clutter your menu and might even prevent you from running the batch file.) Now, let's take a closer look at how to create the version of EASYMENU.BAT shown in Figure A.

A sample menu screen

EASYMENU.BAT begins with the command

```
@echo off
```

which prevents DOS from echoing commands to the screen as it executes them. The next command in the batch file,

```
cls
```

clears your screen. This ensures that no extra text will appear on your menu screen and that you'll be able to position your choices precisely.

After the first two lines in the batch file prepare your screen, you're ready to use ECHO commands to display your menu, one line at a time. EASYMENU.BAT repeats the command

```
echo.
```

seven times to leave seven blank lines at the top of the screen. (Of course, you can repeat the *echo.* command to leave as many lines as you like.) When you type *echo.*, be sure that you leave no space between *echo* and the period.

The next section of EASYMENU.BAT displays the text of your menu:

```
echo       To do this:                  Enter:
echo.
echo       Create/edit a document       go 1
echo       Format a 1.2 Mb diskette     go 2
echo       Format a 360 Kb diskette     go 3
echo       Quit the menu                go 4
```

When you're creating the menu, remember that the word *echo* won't be displayed onscreen, so the "To do this" column will really begin four characters to the right of where it appears in the batch file. With the DOS 5 Editor, you can use tabs to position the columns of the menu.

The next section of EASYMENU.BAT displays two blank lines, then sets a special prompt:

```
echo.
echo.
prompt [Ctrl]P[Alt][255]   Enter your choice:
```


INSIDE DOS™

Inside DOS (ISSN 1049-5320) is published monthly by The Cobb Group.

Prices
Domestic ..... $49/yr. ($6.00 each)
Outside U.S. ..... $69/yr. ($8.50 each)

Phone
Toll free ..... (800) 223-8720
Local ..... (502) 491-1900
FAX ..... (502) 491-8050

Address
You may address tips, special requests, and other correspondence to:

The Editor, *Inside DOS*
9420 Bunsen Parkway, Suite 300
Louisville, KY 40220

For subscriptions, fulfillment questions, and requests for bulk orders, address your letters to:

Customer Relations
9420 Bunsen Parkway, Suite 300
Louisville, KY 40220

Postmaster
Second class postage is paid in Louisville, KY. Send address changes to:

Inside DOS
P.O. Box 35160
Louisville, KY 40232

Back Issues
To order back issues, call Customer Relations at (800) 223-8720. Back issues cost $6.00 each, $8.50 outside the U.S., and you can pay with MasterCard, VISA, Discover, or American Express, or we can bill you. Please identify the issue you want by the month and year it was published. Customer Relations can also provide you with an issue-by-issue listing of all the articles that have appeared in *Inside DOS*.

Staff

Editor-in-Chief ..... Suzanne Thornberry
Contributing Editors ..... Van Wolverton
David Reid
Editing ..... Polly Blakemore
Linda Watkins
Production ..... Margueriete Stith
Design ..... Karl Feige
Publications Manager ..... Elayne Noltemeyer
Managing Editor ..... Jim Welp
Circulation Manager ..... Brent Shean
Publications Director ..... Linda Baughman
Editorial Director ..... Jeff Yocom
Publisher ..... Douglas Cobb

You create the blank lines as you did earlier, typing *echo* followed immediately by a period. But the PROMPT command is a bit trickier. The challenge is an aesthetic one: The menu will look nicer if the prompt lines up under the first column of choices, but you can't begin a PROMPT command with a space or tab.

To work around this limitation, you can use a special space character at the beginning of the prompt. You do this by typing *prompt*, a space, then pressing [Ctrl]P to tell the DOS 5 Editor that you want to insert a special character. Now, to create the special character, hold down the [Alt] key while you type 255 on the numeric keypad. Next, use the same number of spaces or tabs as you used for the choices in column one and press [Spacebar] two extra times. Adding the extra spaces will compensate for the length of the PROMPT command. Remember, the previous columns began with the ECHO command, which is two characters shorter. Once you've spaced over, you can type the phrase *Enter your choice*, followed by a colon and a space. EASYMENU.BAT will replace your standard system prompt (C:\>) with this phrase. You'll be able to type commands here just as you would at the standard prompt.

Creating GO.BAT

Now that you've created a batch file to present the menu, you're ready to create GO.BAT, which will run the selection you choose. GO.BAT will read the number you type after *go* and then go to the corresponding label. As you'd expect, the instructions under each label will run the program or command indicated on the menu. In fact, you can think of GO.BAT as a collection of smaller batch files—one under each label. Although GO.BAT can seem a little confusing at first glance, its components are really very simple. Let's take a look at the GO.BAT that corresponds to our sample EASYMENU.BAT.

A sample GO.BAT

Figure B shows the GO.BAT file that runs the applications in our sample EASYMENU.BAT. You customize GO.BAT by substituting the commands that run each of your selections under the corresponding labels.

Now, let's look more closely at GO.BAT to see how it's structured. Along the way, we'll show you techniques that can make your menu system more flexible. Like most of our batch files, GO.BAT begins with the line *@echo off*, which prevents DOS from displaying commands as it executes them. The next line,

```
goto %1
```

is the mechanism that allows you to run the menu selection. This GOTO command simply sends DOS to the label 1, 2, 3, or 4—whichever corresponds to the number you typed after *go* at the menu prompt. To accomplish this, we use the %1 symbol to stand for the first parameter you typed after the batch file name.

Figure B

```
@echo off
goto %1
:1
c:
cd \docs
word
goto reset
:2
c:
format a:
goto reset
:3
c:
format a: /f:360
goto reset
:4
c:
prompt $p$g
cls
echo.
echo.
echo To return to the menu system, type
echo EASYMENU, then press [Enter].
pause
goto end
:reset
c:
cd \
easymenu
:end
```

GO.BAT runs the applications listed by EASYMENU.BAT.

Running a program

The :1 label contains the commands that run Microsoft Word:

```
:1
c:
cd \docs
word
goto reset
```

Although you could run Word simply by typing *word*—provided the C:\WORD directory is on your path—we've added a few extra commands to customize the way we run Word. First, we change to the C: drive and the C:\DOCS directory, which is where we keep our Word documents. Then, we run Word with the command *word*. When the batch file launches Word, you'll see a list of the documents in the C:\DOCS directory whenever you issue the Open... command from Word's File menu. (In the article "Running Programs and Loading Files within a Batch File," which begins on page 5, we explain some techniques you can

use to make GO.BAT more flexible in running applications.) When you exit Word, GO.BAT will go to the :RESET section, which we'll explain later.

Two formatting options

The second and third menu selections allow you to format a 5.25" diskette in a high-density drive. The section under label 2,

```
:2
c:
format a:
goto reset
```

simply formats a high-density 5.25" diskette to 1.2 Mb. The instructions under label 3,

```
:3
c:
format a: /f:360
goto reset
```

format a 5.25" diskette to 360 Kb. Again, both of these sections will go to the :RESET label when you've finished formatting diskettes. (The article "Choosing the Right Format Option for the Right Diskette," which appears on page 10, gives you more details on these formatting options.)

Quitting the menu

The instructions under label :4 allow you to exit from the menu to a "plain" DOS prompt:

```
:4
c:
prompt $p$g
cls
echo.
echo.
echo To return to the menu system, type
echo EASYMENU, then press [Enter].
pause
goto end
```

The first line restores the normal system prompt, which shows the path and greater-than sign. Then, the *cls* instruction clears the menu from the screen. The next ECHO commands display two blank lines, followed by a message reminding you how to get back to the menu. After you read the mesage, the PAUSE command will allow you to continue by pressing a key. When you press a key to continue, the next instruction clears the screen. Unlike options 1 through 3, option 4 doesn't return to the menu. Instead, it goes to the last line of the batch file, *:end*. By going to the :END label, the batch file will skip over the instructions in the :RESET section. Remember, once you choose option 4, you won't automatically return to the menu; you'll have to type *easymenu* to run it again.

Returning to the menu

As we've seen, options 1 through 3 jump to the :RESET label when you quit the application. The :RESET section contains instructions to reset your menu system, as shown below:

```
:reset
c:
cd \
easymenu
:end
```

To ensure that you always begin in the same directory, :RESET changes to the C: drive and the root directory. Then, the line *easymenu* runs EASYMENU.BAT, again displaying the menu.

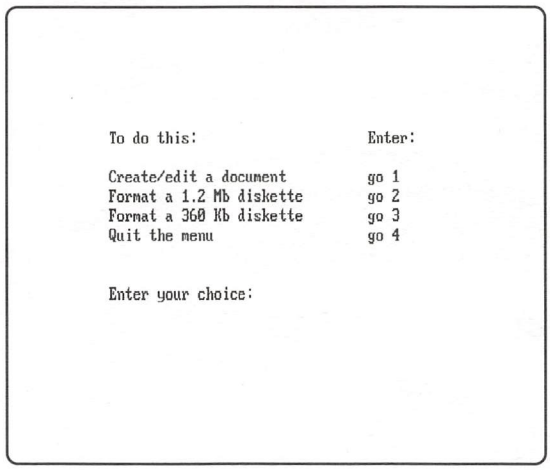
Using the menu system

After looking at how EASYMENU.BAT and GO.BAT work, you should have a pretty good idea of how to use them. You can display the menu from the DOS prompt by entering the following command:

```
C:\>easymenu
```

When you do, DOS will display the menu, as shown in Figure C.

Figure C



When you run EASYMENU.BAT, you'll see the choices you placed in the GO.BAT batch file.

If you'd like to see the menu every time you use your computer, just place the *easymenu* command at the end of your AUTOEXEC.BAT file. (As always, be sure to edit your AUTOEXEC.BAT with DOS 5's Editor or another ASCII-compatible word processor.)

Once you've displayed the menu, you simply follow the instructions you listed. For example, if you're using our versions of EASYMENU and GO.BAT, you'd enter *go 2* to format a 1.2 Mb diskette. When you answer *N* to

the question *Format another (Y/N)?*, DOS will return to GO.BAT, read the *goto reset* instruction, and redisplay your menu.

Notes

In our example, we've used numbers for the applications. But the EASYMENU system isn't limited to single-digit numbers. You can use code names (such as *wd*), words (such as *add*), or longer numbers (such as *101*). Of course, you'll need to use the same codes, words, or numbers as labels when you create GO.BAT.

You can also use EASYMENU.BAT with a batch file that sets screen colors. Just be sure to run EASYMENU *after* the batch file that sets colors. If you run EASYMENU from the AUTOEXEC.BAT file, use the CALL command to run your color batch file, then place the line *easymenu* after it.

Conclusion

In this article, we've shown you how to create a simple menu system with two batch files. EASYMENU.BAT will display the selections you echo to the screen. You run the selections with GO.BAT by typing *go* followed by the number (or name) you assigned to the menu selection.

The following article demonstrates a technique for creating menus by redefining function keys:

"Setting Up a Menu System with Simple DOS Commands," November 1990

For information on ordering back issues, see the masthead on page 2 of this issue.

BATCH FILE TIPS

Running programs and loading files within a batch file

Many DOS users create batch files to run their favorite programs. You might create several batch files, each one running a different program. Or, you might create one "master" batch file, like the GO.BAT file we created in the article "Creating a Simple Menu System" on page 1. Whether you create a master batch file or individual batch files for running programs, you can enhance the batch files so that they can load a file as they run the program. In this article, we'll show you how to automate loading a particular file each time you run the batch file and how to tell your batch file to load any file you specify. We'll start by looking at how you can load a particular file to carry out a common task.

Automating a common task

Do you find that you frequently update the same document, spreadsheet, or database file? If you do, you can save time by loading the file you want to update when you run the program. Most applications allow you to do this simply by typing the path (if it's not the current path) and name of the file after the command that runs the application. You can make the task of running the program and loading that particular file even easier by adding the command to a batch file.

For example, suppose you frequently use Microsoft Word to update a file of names and addresses. Let's also suppose that this file is named NAMES.DOC and is stored in your C:\DOCS directory. You can create a batch file called UPDATE.BAT to load NAMES.DOC automatically, as we've done below:

```
c:
cd \docs
word names
```

(Word assumes that the file has the DOC extension)

This technique works in master batch files as well as in standalone batch files. For example, you can specify a filename in GO.BAT by changing the section under label :1, as shown below:

```
:1
c:
cd \docs
word names
goto reset
```

Loading the filename you type

So far, we've shown you how to set up a batch file to load the same file every time you run an application. But what if you need to specify *different* files each time you run an application? You can do this easily from the command line by typing the name of the application, followed by the name of the file. However, when you run the application through a batch file, you won't

necessarily be able to load a file. For example, suppose you've created W.BAT, a batch file that changes to your C:\DOCS directory and runs Microsoft Word:

```
c:
cd \docs
word
```

This version of W.BAT won't allow you to specify a file as you run Word. However, you can easily revise this file so that you can type the name of the file you want to edit after typing the *w* batch file name. Simply add the symbol for the first parameter—%1—to the WORD command in the batch file, as we've done below:

```
c:
cd \docs
word %1
```

Now, whenever you run W.BAT, Word will check to see if you typed a filename as the first parameter after *word*. For example, you could load the file C:\DOCS\ZEBRA.DOC as you run W.BAT with the following command:

```
C:\>w zebra
```

Notice that Word can locate the ZEBRA.DOC file because W.BAT changes to the C:\DOCS directory before it runs Word.

Adding flexibility to GO.BAT

You can also edit GO.BAT so that you can load any file you specify with the GO command. However, you'll need to use the %2 symbol to stand for the name of the file you want to load. That's because GO.BAT already uses %1 in the statement

```
goto %1
```

Returning to our example, you can edit the instructions under label :1 of GO.BAT so that you can load any file you specify. Just add the %2 parameter to the WORD command, as shown below:

```
:1
c:
cd \docs
word %2
goto reset
```

With this version of GO.BAT, you can load the file C:\DOCS\ZEBRA.DOC as you run Word by entering the following command:

```
Enter your choice: go 1 zebra
```

Conclusion

In this article, we've shown you how to create a batch file to load documents you work on frequently. Or, for more flexibility, you can use parameters so that you can specify any document you want to load as you run the batch file.

For more tips on creating flexible batch files, see the following article:

"Creating Batch Commands That Operate on Variable Data," July 1990

For information on ordering back issues, see the masthead on page 2 of this issue.

COPY WORKAROUND

The BACKUP command lets you copy a large file onto multiple diskettes

Have you ever tried to copy a large file to a blank, formatted diskette, only to be met with the message

```
Insufficient disk space
0 file(s) copied
```

If you're lucky, you can switch to a high-density diskette and try again. But if you don't have a high-density drive, or if your file requires even more room than a high-density diskette can provide, you'll continue to be frustrated in your attempts to make the copy. You might resort to cutting the file into smaller chunks from within your application, or you might even try to download the file from your PC through a modem or serial connection, which may require additional hardware and software.

When you need to copy large files, you don't need to invest in hardware or software or spend time tediously chopping the file into manageable pieces. Instead, you can use DOS' BACKUP command to copy the file onto two or more diskettes. Of course, when you use the BACKUP command to copy a large file, you'll have to restore the file to a hard disk to read it again.

Before we explain the procedure, we need to caution you about the drawbacks of the technique. Unlike true

copies made with the COPY or XCOPY command, you won't be able to use the backed-up file directly from the diskettes. Also, you'll have to be sure to record the name of the directory that originally contained the file, or else the RESTORE command won't work. Despite these drawbacks, BACKUP and RESTORE offer a way to copy large files without investing in a third-party utility.

In this article, we'll explain how to use the BACKUP and RESTORE commands to copy a large file. If you've never tried DOS' backup feature, this article will help you get started with backing up and restoring files.

What you'll need

Before we show you the technique for backing up a large file, let's review what you'll need. To make sure that the process goes smoothly, we suggest that you begin by gathering enough formatted diskettes to hold all of the file. For example, if you're backing up a 2 Mb file, you'll need just two 1.44 Mb, 3.5" diskettes, or two 1.2 Mb, 5.25" diskettes. But you'll need six 360 Kb, 5.25" diskettes to back up the same file. (If you're transferring the file to a PC that runs a compatible version of DOS, make sure your diskettes are compatible in size and capacity to the PC you'll use to restore the file. The article "Choosing the Right Format Option for the Right Diskette," which begins on page 10, provides more information on choosing the right diskette.) When you've prepared the diskettes, label them with the path and name of the file you copied and then number them. Later, you'll need to restore the diskettes in the same order as BACKUP created them.

Using the BACKUP command

Once you've formatted your diskettes, backing up the file is a fairly straightforward process. First, change to the directory that contains the file you want to back up. Then, type *backup*, followed by the name of the file you want to back up, then the name of the drive you want to receive the backup.

For example, suppose you want to copy the file HUGEFILE.DAT, which is in your C:\DATA directory. You can find the size of HUGEFILE.DAT by changing to the C:\DATA directory and entering the command *dir hugefile**. The resulting directory information shows the file's size in bytes:

```
HUGEFILE   DAT      2625121          09-28-92          1:44p
```

As you can see, HUGEFILE.DAT contains 2,625,121 bytes. Let's also suppose that you'll back up your file to 1.44 Mb, 3.5" diskettes in your A: drive. Since 2,625,121 is roughly equal to 2.6 Mb, you'll need to use two 1.44 Mb diskettes to back up the file. When you've prepared the diskettes, place the first diskette in the A: drive. Then, change to the C:\DATA directory and enter the following command:

```
C:\DATA>backup hugefile.dat a:
```

After you enter this command, DOS will present a stern message:

```
Insert backup diskette 01 in drive A:

WARNING! Files in the target drive
A:\ root directory will be erased
Press any key to continue...
```

This is your last chance to save the files on your diskette before BACKUP overwrites them with HUGEFILE.DAT. If you want to protect the data on the diskette, press [Ctrl]C or [Ctrl][Break] to quit the BACKUP command. But, if your diskettes are blank or the information on them is disposable, press a letter, number, space, or [Enter] to continue the operation. As DOS backs up the file, it will present the message

```
***Backing up files to drive A:***
Diskette Number: 01

\DATA\HUGEFILE.DAT
```

In a few moments, the BACKUP command will fill up the first disk with a portion of HUGEFILE.DAT. When it does, DOS will present the message

```
Insert backup diskette 02 in drive A:
```

followed by another warning about erasing the files on the target diskette. At this point, you remove the first diskette and insert the second diskette. (Be sure to remove the first diskette *before* you continue; otherwise, BACKUP will overwrite the first portion of the file.) Then, you press a key to continue backing up the file. As DOS backs up the file, it will display the message

```
***Backing up files to drive A:***
Diskette Number: 02

\DATA\HUGEFILE.DAT
```

When the backup is complete, DOS will return you to the C:\DATA> prompt. You now can remove the diskette from the drive.

If you issue the DIR command for each diskette, you'll find that neither one contains a file named HUGEFILE.DAT. Instead, both diskettes contain files named BACKUP and CONTROL. On the first diskette, the two files have the extension 001, as you can see in the directory listing below:

```
BACKUP     001      1457152          3:07p
CONTROL    001          243          3:07p
```

The files on the second diskette are similar. However, the BACKUP.002 file is smaller than BACKUP.001—BACKUP didn't need to use all the space on the diskette for the rest of the file:

```
BACKUP     002      1167969        3:08p
CONTROL    002          243        3:08p
```

Restoring the file

Once you've backed up the file, you can restore it to your hard disk or to the hard disk on another computer. The process is fairly straightforward, but you'll need to bear a few things in mind:

- You must restore the file to the same directory as it came from. If you're copying the file to another machine, you'll have to create a directory of the same name—unless, of course, the computer already has such a directory.
- If a file of the same name already exists in that directory, the BACKUP command will overwrite it. (We'll show you a workaround for this problem in the "Notes" section.)
- If you're restoring a file to another machine, that PC also should run DOS 5. If it doesn't, you can make a boot diskette, copy the RESTORE command to it, and boot your colleague's PC with DOS 5 *before* you restore the file to it. We describe this technique in the sidebar "Booting with DOS 5 before Restoring a File" on page 9.

When you're ready to restore the file, you'll need to give the RESTORE program several pieces of information. To use the command, you type *restore*, a space, the name of the drive that contains the backup, a space, then the drive, path, and name of the file you want RESTORE to re-create.

With this in mind, let's look at how you'd restore HUGEFILE.DAT. Suppose you want to give a colleague a copy of the file, so you'll be restoring it to his or her computer, which also runs DOS 5. (Again, if your colleague uses a previous version of DOS, see the sidebar "Booting with DOS 5 before Restoring a File.") Let's also suppose that this colleague doesn't have a directory named C:\DATA. You begin by creating a \DATA directory on his or her computer. You can do this by changing to the root directory on the C: drive, then entering the command

```
C:\>md data
```

Now, place the first backup diskette in the A: drive of your colleague's computer. Next, enter the following command:

```
C:\>restore a: c:\data\hugefile.dat
```

DOS will present the message

```
Insert backup diskette 01 in drive A:
Press any key to continue...
```

Since you've already inserted the diskette, press a character key or [Enter] to begin the restoration. The messages generated by the RESTORE command are similar to the ones you saw when you backed up the file. As it restores BACKUP.001, DOS will display a message telling you that it's restoring files from drive A:. When DOS locates the HUGEFILE.DAT file, it will display *\DATA\HUGEFILE.DAT*. Then, DOS will prompt you to insert the second backup diskette in drive A:. Remove the first diskette, insert the second one, then press a key. DOS will link the information in BACKUP.002 to BACKUP.001, re-creating HUGEFILE.DAT in your colleague's C:\DATA directory.

Common RESTORE errors

As long as you're careful to specify the same path and name for the file in the RESTORE command, you should have no trouble restoring the file you backed up. But minor differences in file or path names can disrupt the RESTORE procedure. (That's why we recommend that you label your backup diskettes with the complete path and name of the file you backed up.) Sometimes when you're restoring a file, DOS will quickly read the first diskette and then prompt you to insert the second one. Then, DOS will pause for a moment and display the message

```
WARNING! No files were found to restore
```

If this happens, check to see that you specified the same path name and filename as you did when you backed up the file.

If you accidentally insert a diskette that doesn't contain the BACKUP.001 file, you'll see the message

```
WARNING! Diskettes out of sequence
Replace diskette or continue if OK
```

Should this happen, insert the appropriate backup disk and press a key. As long as the diskette you inserted contains the BACKUP.001 file, you should be able to continue without any problems.

Notes

What if you need to restore a file to your hard disk, but you already have a more recent copy of the file in the same directory? Normally, the RESTORE command will overwrite the file. Since you don't want to destroy the more recent copy of the file, you'll press *N* to quit the RESTORE command.

If you really need to work with the older version of the file, simply rename the version on your hard drive. For example, you could issue the command

```
C:\DATA>ren hugefile.dat hugefil1.dat
```

Changing just one character in the filename will allow you to restore the backed up HUGEFILE.DAT to your C:\DATA directory, but you'll still preserve all the information in HUGEFIL1.DAT, the newer version.

Conclusion

In this article, we've shown you how to use the BACKUP and RESTORE commands to copy a large file to two or more diskettes. Along the way, we've explained the basics of using DOS' backup feature. In future articles, we'll show you more techniques for backing up all your important data, including how you can save time by letting BACKUP format your diskettes for you.

Booting with DOS 5 before restoring a file

In the article "The BACKUP Command Lets You Copy a Large File onto Multiple Diskettes" on page 6, we showed you how to back up a file from your PC and restore it onto a colleague's PC—providing you're both using DOS 5. With a bit of preparation, you can even restore a file to a PC that normally uses an older version of DOS. The trick is to boot your colleague's PC with a special diskette that you create in your PC.

Creating the boot diskette

You begin with a diskette size and capacity that you both can use. (We discuss how to choose diskettes in "Choosing the Right Format Option for the Right Diskette" on page 10.) Then, you use the FORMAT command with the /S switch to transfer COMMAND.COM and two hidden system files to the diskette. To do this, you type *format*, a space, the letter of the diskette drive you're formatting, a colon, a space, a forward slash, and an *s*. For example, to format a diskette in the A: drive as a bootable diskette, you enter

```
C:\>format a: /s
```

Once you've formatted the diskette with the /S switch, copy the RESTORE.EXE file from your DOS directory to the diskette with the following command:

```
C:\>copy \dos\restore.exe a:
```

That's all there is to preparing the DOS 5 boot diskette. Now, let's take a brief look at how you can use it.

Using the boot diskette

As with any boot diskette, you simply place the diskette in the A: drive of your colleague's computer, then press [Ctrl][Alt][Del] to restart the system. You should see the A> prompt.

Now is the time to take an important precaution. To make sure that the newer version of DOS can read from the hard drive, change to the C: drive and enter *dir*. If you see strange characters instead of filenames, *immediately* remove the boot diskette from the diskette drive and reboot the system without it. In this case, DOS 5 can't read the hard drive, and it could destroy data—or even damage the hard disk—if you try to restore a file using the newer version. (This usually happens with special versions of PC-DOS created by computer manufacturers.)

On the other hand, if you can display a directory of the files on the hard drive, you should have no problems backing up or restoring a file. If you're restoring a file, you can now create a directory of the same name as the one that contained the original file. (You can skip this step if such a directory already exists.)

Once you've verified that the C: drive is readable and created a directory (if needed), switch back to the diskette drive. For example, if you've booted from the A: drive, type *a:*, then press [Enter] to change back to the boot diskette. Now you can issue the RESTORE command from the diskette. (Remember, you must issue the version of the command on the diskette—*not* the version on the hard drive.) DOS will prompt you to insert a diskette, so you'll now need to remove the boot diskette. From this point on, you use the same techniques as described in the article "The BACKUP Command Lets You Copy a Large File onto Multiple Diskettes."

For more information on booting with a diskette, see the following article:

"Van Wolverton: Protecting Your Data and Equipment from Disaster," July 1992

For information on ordering back issues, see the masthead on page 2 of this issue.

Choosing the right format option for the right diskette

In the preceding article, "The Backup Command Lets You Copy a Large File onto Multiple Diskettes," we mention that you should estimate how many diskettes you'll need to hold a file before you begin the backup process. To do this, you need to know the capacity of your diskettes, as well as the capacity of your diskette drive. If you plan to give the backup to a colleague, you'll also need to know the capacity of his or her diskette drive. Once you have this information, you'll need to choose diskettes of the *lowest* capacity for the job.

For example, suppose you have a 5.25" diskette drive that can read 1.2 Mb diskettes, while your colleague has a 5.25" diskette drive that can read only double-density (360 Kb) diskettes. In this case, you'll need to use diskettes formatted to 360 Kb for the backup. In fact, we suggest that you first format the diskettes in your colleague's computer to ensure that the drive can read them, then back up the file you want to share.

Now that we've shown you why diskette density can be important, let's look at the topic a little more closely. We'll begin by reviewing the capacity of some common diskettes. Then, we'll show you how diskette capacity and drive capacity are factors in formatting the diskettes. We'll also demonstrate the commands you use to format lower-capacity diskettes in a higher-capacity drive.

Capacity: how much data a diskette can hold

As you know, diskettes for today's PCs come in two physical sizes: 5.25" and 3.5". You can tell which size diskette you need simply by looking at the slot on your drive door. Unfortunately, it's not as easy to tell what capacity diskettes the drive can use. Table A shows the choices you have for most diskette drives. As you can see, the capacity tells you how many kilobytes (Kb) or megabytes (Mb) of data can fit on the formatted diskette. Throughout the table, DS stands for double-sided. Diskettes manufactured in recent years are almost always double-sided.

Table A

Size	Density	Capacity
5.25"	Double (DS, DD)	360 Kb
5.25"	High (DS, HD)	1.2 Mb
3.5"	Double (DS, DD)	720 Kb
3.5"	High (DS, HD)	1.44 Mb

Most personal computers can use one or more of the diskette capacities listed here.

When you buy diskettes packaged in boxes, each diskette is labeled with the manufacturer's name, the density of the diskette, and its capacity. Often, the density is abbreviated as shown in Table A. For example, a double-sided, double-density diskette is indicated by the code *DS, DD*. Most manufacturers' labels also show the formatted capacity of the diskette, also listed in Table A. A few manufacturers may list a slightly higher number for the capacity. For example, a double-sided, high-density 5.25" diskette may list 1.6 Mb as its capacity, rather than 1.2 Mb. These higher numbers mean that the manufacturer is listing the capacity of an unformatted diskette. DOS uses a portion of that capacity when you format the diskette.

By the way, high-density 3.5" diskettes are usually easy to spot. Almost all of them display the stylized HD symbol shown in Figure A.

Figure A

High-density 3.5" diskettes sport this special symbol.

Rules of thumb

Here are a few rules of thumb for using diskettes of various capacities:

- If your diskette drive can read formatted, high-density diskettes, it should also be able to read formatted, double-density diskettes.
- If you have a double-density drive, you waste money if you regularly buy high-density diskettes. The drive will only format them to 360 Kb (or 720 Kb, for 3.5" drives), and you'll pay extra for the capacity you can't use.
- If you have a high-density diskette drive, don't try to format double-density diskettes in it—at least not without using a special form of the FORMAT command. If you attempt this, you'll probably hear strange sounds as DOS attempts to format the diskette to a higher capacity than it was designed for. After several minutes, FORMAT will quit. You might think that the diskette is formatted, but diskettes formatted above their capacities aren't reliable.

Formatting lower-density diskettes

If you have a high-density drive, you can use the FORMAT command with special switches to format double-density diskettes. While these switches will work on most systems, they can be incompatible with some versions of PC-DOS and with some diskette drives or controllers. So, if you've never used these FORMAT switches before, we suggest that you try them on some sample diskettes, then try to copy files to and from the diskettes as a test. If your tests are successful, you should be able to use double-density diskettes formatted in your high-density drive with no trouble. If you have problems in your tests, however, you should either stick to high-density diskettes formatted with the standard FORMAT command or use preformatted double-density diskettes.

With those caveats in mind, let's take a look at these special versions of the FORMAT command. Suppose you have a high-density diskette drive and you need to format a 5.25" double-density diskette to its 360 Kb capacity. To do this, you add the /F:360 switch to the FORMAT command. So, to format the 5.25" diskette to 360 Kb, you'd enter

```
C:\>format a: /f:360
```

The command for formatting lower-density diskettes in a 3.5" high-density drive is similar. To format a 3.5" diskette to 720 Kb capacity, you add the /F:720 switch to the FORMAT command. For example, to format a 3.5" double-density diskette in your B: drive, you enter the following command:

```
C:\>format b: /f:720
```

Conclusion

In this article, we've reviewed the four most common diskette capacities used by today's personal computers. We've outlined some of the potential incompatibilities among the formats. We also showed you some commands for formatting double-density diskettes in a high-density drive.

For more information on formatting diskettes, see the following article:

"Formatting Diskettes: It's Not So Simple Anymore," January 1991

For information on ordering back issues, see the masthead on page 2 of this issue.

LETTERS

Turning off extended break checking

Is there a way to prevent people from stopping a batch file by pressing [Ctrl][Break] or [Ctrl]C? I thought I read that you could do this through AUTOEXEC.BAT, but I can't remember the exact command.

Ken Bectel
Lake Charles, Louisiana

You can tell DOS not to check for the break keys—[Ctrl][Break] and [Ctrl]C—by entering a BREAK command from the command line, the AUTOEXEC.BAT, or any batch file. The command

```
break off
```

limits the use of the break keys. As you might expect, you can turn break checking back on with the command

```
break on
```

If you'd like to suspend break checking just for the duration of a batch file, you can place the *break off* command at the beginning and the *break on* command at the end of the batch file.

You should note, however, that the BREAK command doesn't provide an absolute way of preventing people from quitting programs or batch files. Even with BREAK turned off, DOS still will act on the break keys whenever it writes to the screen or to the printer. With BREAK turned on, DOS will also check to see if you've pressed the break key whenever it reads from or writes to the disk.

Automatically typing DISKREPT.RPT

I created the DISKREPT.BAT and DISKREPT.BAS files as described in the article "Determining the Amount of Disk Space Each of Your Directories Consumes" in the May 1992 issue of *Inside DOS*. Although the utility works fine as you presented it, I added a command to type DISKREPT.RPT to the end of my DISKREPT.BAT file. This saves me the trouble of issuing the TYPE command after I run the program.

I also have a question about the utility: Is there any way to prevent the QBasic Editor screen from appearing when DISKREPT.BAS runs?

Francia Ruppen
King of Prussia, Pennsylvania

Thanks for sharing your enhancement to DISKREPT.BAT with us. Figure A shows the new version of the file, which includes the TYPE command, as you suggested. Using the MORE filter ensures that you'll be able to read even long disk reports.

Figure A

```
@echo off
echo Space usage report for directory %1 >
c:\diskrept.txt
dir /s %1 >> c:\diskrept.txt
qbasic /run c:\diskrept
del c:\diskrept.txt
type c:\diskrept.rpt | more
```

Francia Ruppen added the TYPE command to further automate DISKREPT.BAT.

As for your question, we're sorry to report that there's no way to prevent the QBasic Editor screen from appearing. DOS must open the QBasic Editor to run the DISKREPT.BAS program.

Placing subdirectories on the path

I need to place both my C:\WORD and C:\WORD\MYWRITE directories in the PATH statement in my AUTOEXEC.BAT file. Do I need to list these directories separately on the path or will the longer reference (C:\WORD\MYWRITE) tell DOS to look in the C:\WORD directory as well? If not, is there some other way to combine the directory entries? I'm concerned that I may be pushing the limits on my PATH command.

J. D. Meekma
Mabank, Texas

You're right to be concerned about the length of your PATH command. DOS will read only up to 127 characters in the command line. When you subtract the minimum of five characters required to type *path* and a space, you're left with just 122 characters to list the directories on your path. In fact, you might not even be aware that you've entered more than the limit for the path. DOS will simply truncate the command—perhaps leaving off some directories you intended to place on the path. In this case, you won't be able to run the programs, commands, or batch files in that directory without first changing to the directory. If you try to run them without changing to the directory, DOS will present the message

```
Bad command or file name
```

You can confirm the path as the problem by changing to the directory that contains the file you want to run, then entering the filename. If the program, command, or batch file runs, you'll know that the path was the cause. (If it still doesn't run, you may have mistyped the command name or the command file may be missing or damaged.) You can check to see what directories are on the path by typing *path*, then pressing [Enter].

Since DOS limits the length of the path, it would be nice if you could use a kind of shorthand to name both a parent directory (such as C:\WORD) and its child directory (such as C:\WORD\MYWRITE) in the path. Unfortunately, you have to list each directory name separately. In the case you've described, the PATH statement in your AUTOEXEC.BAT might look something like this:

```
path c:\;c:\dos;c:\word;c:\word\mywrite
```

As you can see, you must type the full form of each directory name and separate the names with semicolons.

If DOS is cutting off your path, you have a couple of options. First of all, decide whether you really need a directory on your path. You only need to include a directory if it contains a batch file, command, or executable file that you frequently run. For example, you could safely delete the C:\WORD\MYWRITE directory from your path if it contains only documents. On the other hand, if you keep just one or two batch files, commands, or executable files in a directory, consider copying the file(s) to a directory that you must include on your path anyway. For example, if you keep a batch file named LETTERS.BAT in your C:\WORD\MYWRITE directory, you could move the batch file to your C:\WORD directory and delete the MYWRITE subdirectory from your path.